Modeling Science: From Free Fall to Chaos

Versión en castellano

Wolfgang Christian y
Francisco Esquembre

PARTE 1

Modeling With *Easy Java Simulations*

Capítulo One

Introducción a Easy Java Simulations

Un buen ejemplo es el mejor sermón. *Benjamin Franklin*

Este capítulo nos ofrece una visión general de *Easy Java Simulations* (de forma abreviada *EJS*), el programa de modelado de alto nivel y herramienta de autor que utilizaremos para construir nuestros modelos y ejecutarlos de forma que podamos estudiar su comportamiento. Para proporcionar una perspectiva del proceso de modelado, primero cargaremos, inspeccionaremos y pondremos en funcionamiento una simulación existente de un oscilador armónico simple. Posteriormente, modificaremos esta simulación para mostrar cuál es el papel del usuario de *EJS* en el proceso de modelado y cómo realmente reduce la mayoría del trabajo de programación necesario.

1.1. SOBRE Easy Java Simulations

El modelado por ordenador está íntimamente ligado a la simulación por ordenador. Un modelo es una representación conceptual de un sistema físico y sus propiedades y el modelado es el proceso por el que construimos dicha representación. El modelado por ordenador necesita (1) una descripción y análisis del problema, (2) la identificación de las variables y los algoritmos, (3) la implementación de una plataforma específica de hardware y software, (4) la ejecución de lo implementado y el análisis de los resultados, (5) el refinamiento y generalización, y (6) la presentación de resultados. Una simulación es una implementación de un modelo de forma que nos permita probarlo bajo diferentes condiciones con el objetivo de aprender sobre su comportamiento. La aplicación de los resultados de la simulación al sistema (físico) real depende de lo bien que el modelo describa la realidad. El proceso de concebir modelos más generales y más ajustados es el objetivo primordial de la ciencia.

La implementación de un modelo y la visualización de sus salidas requiere que programemos un ordenador. Programar puede ser divertido, ya que nos da un completo control de cada detalle visual y numérico del mundo

de la simulación. Sin embargo programar es también una tarea técnica que puede intimidarnos. Esta barrera técnica puede, sin embargo, rebajarse si usamos la herramienta apropiada. *Easy Java Simulations* es una herramienta de modelado que ha sido diseñada para permitir a los científicos, y no sólo a los informáticos, crear simulaciones en Java. *EJS* simplifica esta tarea desde el punto de vista técnico y conceptual.

EJS proporciona una simple pero poderosa estructura conceptual para construir simulaciones. La herramienta ofrece una secuencia de paneles de trabajo que usamos para implementar el modelo y su interfaz gráfica de usuario. *EJS* automatiza tareas como las de resolución de ecuaciones diferenciales ordinarias (con diferentes *hilos* – threads, en inglés – de Java) y de animación. La comunicación de bajo nivel entre el programa y el usuario final que tiene lugar en el momento de la ejecución, incluyendo las acciones que a través del ratón podemos llevar a cabo sobre la interfaz gráfica de la simulación, se logra sin necesidad de una programación de bajo nivel.

Obviamente, parte de la tarea depende todavía de nosotros. Usted es el responsable de proporcionar un modelo para el fenómeno y de diseñar y seleccionar una visualización que muestre los aspectos principales del mismo. Estas tareas de alto nivel están más relacionadas con la ciencia que con la programación. Se le anima, pues, a dedicar su tiempo y esfuerzos a estudiar la ciencia, algo que la máquina no puede hacer. El propósito de este capítulo es demostrar que este modelado por ordenador no sólo es posible, si no que además es relativamente sencillo, con ayuda de *Easy Java Simulations*.

1.2. INSTALACIÓN Y ARRANQUE DEL SOFTWARE

Vamos a empezar instalando y ejecutando *Easy Java Simulations*. *EJS* es un programa en Java que puede ejecutarse bajo cualquier sistema operativo que soporte una Máquina Virtual Java (VM). Ya que Java es una plataforma independiente, la interfaz de *EJS* en Mac OS X, Unix y Linux será prácticamente idéntica a la interfaz de Windows que se muestra en este libro.

Para instalar y ejecutar *EJS* procederemos como sigue:

1. **Instalar el entorno de ejecución de Java.** *EJS* precisa del entorno de ejecución de Java (JRE), en su versión 1.5 o posterior. El JRE puede estar ya instalado en su ordenador, pero, de no ser así, utilice la copia proporcionada en el CD que acompaña a este libro, o mejor, visite la página oficial de Java en <*http://java.sun.com*> y siga las instrucciones de cómo descargar e instalar la última versión.
2. **Copiar *EJS* en su disco duro.** Encontrará *EJS* en un archivo com-

primido ZIP llamado algo así como **EJS_X.x_yymmdd.zip** en el CD del libro. Aquí los caracteres `X.x` se refieren a la versión actual del software y `yymmdd` indica la fecha en que esta versión ha sido creada. (Por ejemplo, puede encontrar algo como **EJS_4.1_081007**.). Descomprima el archivo en su disco duro para crear una carpeta llamada **EJS_X.x** (**EJS_4.1** en este caso). Esta carpeta contiene todo lo necesario para ejecutar *EJS*. [1]

3. **Ejecutar la consola *EJS*.** Dentro de la recién creada carpeta **EJS_X.x**, podemos encontrar un archivo llamado **EjsConsole.jar**. Haciendo doble clic sobre éste ejecutaremos la consola de *EJS* que se muestra en la Figura 1.1.

   Si la consola no se ejecutara al hacer doble clic, abra un terminal del sistema operativo, cambie el directorio de rabajo a **Ejs**, y escriba el comando: `java -jar EjsConsole.jar`. Necesitará cualificar completamente el comando de `java` si éste no está en el PATH de su sistema.

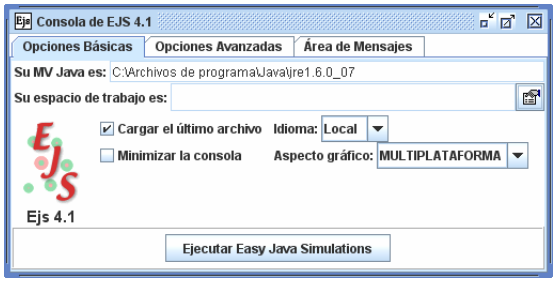

Figura 1.1: Consola de *EJS*.

La consola debería verse en su pantalla como en la Figura 1.1. Esta consola no es parte de *EJS*, sino una utilidad usada para lanzar una o varias copias de *EJS* y llevar a cabo otras tareas relacionadas con *EJS*. La consola muestra por pantalla información del programa y mensajes de error y nos referiremos a ella en alguna que otra ocasión a lo largo de este libro. La consola crea una instancia (copia) de *EJS* al inicio y termina automáticamente cuando se cierra la última instancia de *EJS* en funcionamiento. Otras propiedades de la consola, como su capacidad de procesar colecciones de modelos *EJS*, se describen en los apéndices.

Sin embargo, antes de que la consola de *EJS* se ejecute tras la instalación, aparecerá la ventana que puede ver en la Figura 1.2, donde se le pedirá elegir el directorio de su disco duro que servirá de *espacio de trabajo*.

[1] En sistemas tipo Unix, el directorio **EJS_X.x** podría ser descomprimido con permisos de sólo lectura. En este caso, habilite los permisos de escritura para el directorio **EJS_X.x** y todos sus subdirectorios.

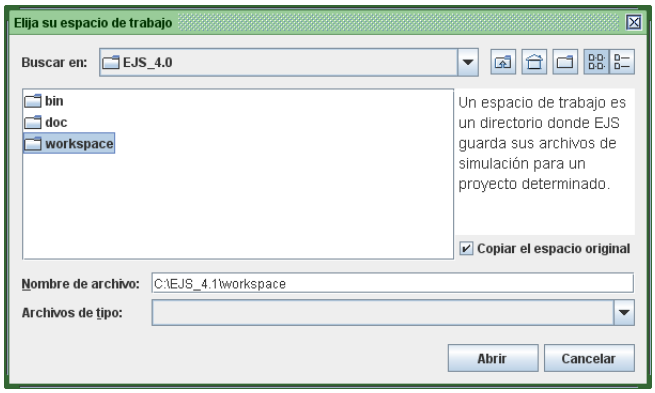

Figura 1.2: Elección del directorio para el Espacio de Trabajo.

EJS utiliza el concepto de espacio de trabajo para organizar su trabajo. El espacio de trabajo es un directorio en su disco duro donde *EJS* almacena los archvos de las simulaciones de un proyecto determinado, pudiendo almacenar un número ilimitado de éstos. Dentro de un espacio de trabajo, *EJS* crea cuatro subdirectorios:

- **config** es el directorio donde *EJS* guarda la configuración determinada por el usuario y otros archivos de opciones.
- **export** es el directorio de destino propuesto por *EJS* cuando se generan archivos listos para su distribución.
- **output** es el directorio usado por *EJS* para situar los archivos temporales generados cuando compilamos una simulación.
- **source** es el directorio donde debe usted colocar todos los archivos necesarios para sus simulaciones (tanto los fuente como los auxiliares).

La primera vez que se ejecuta *EJS*, la consola nos pide que elijamos un directorio para nuestro espacio de trabajo. Éste debe ser un directorio con permiso de escritura, situado en cualquier lugar de nuestro disco duro. Se puede elegir el espacio de trabajo incluido en la distribución, es decir, el directorio **workspace** de la carpeta **EJS_X.x** creada al descomprimir el paquete de *EJS*. Aunque se recomienda crear una nueva carpeta en su directorio habitual destinada a ello. El cuadro de diálogo que le permite elegir el espacio de trabajo tiene una casilla que, si se marca, hará una copia de los archivos de ejemplo que contiene la distribución en el nuevo espacio de trabajo. Deje la casilla marcada y encontrará algunos subdirectorios en el directorio **source** de su espacio de trabajo que contienen simulaciones de muestra. En particular, el directorio **ModelizandoLaCiencia** incluye los modelos de *EJS* de los que se habla en este libro.

Se puede crear más de un espacio de trabajo para diferentes proyectos o tareas. La consola contiene un selector que permite cambiar el espacio de trabajo en uso y *EJS* recordará el actual espacio de trabajo de una sesión a otra o incluso si se reinstala el programa. En el Apéndice **??** se describe cómo configurar y usar *EJS* en una instalación para más de un usuario.

Finalmente, también la primera vez que ejecute *EJS*, el programa le solicitará que introduzca su nombre y filiación. Este paso es opcional aunque recomendado, ya que le ayudará a documentar sus futuras simulaciones. Puede usted introducir o modificar esta información posteriormente a través del icono de opciones de la barra de tareas de *EJS*.

Ahora ya estamos preparados para centrar nuestra atención en la herramienta de modelado *EJS*, que se muestra en la Figura 1.3 con algunas anotaciones. A pesar de presentar una interfaz sencilla, *EJS* contiene todas las herramientas necesarias para el ciclo completo de modelización.

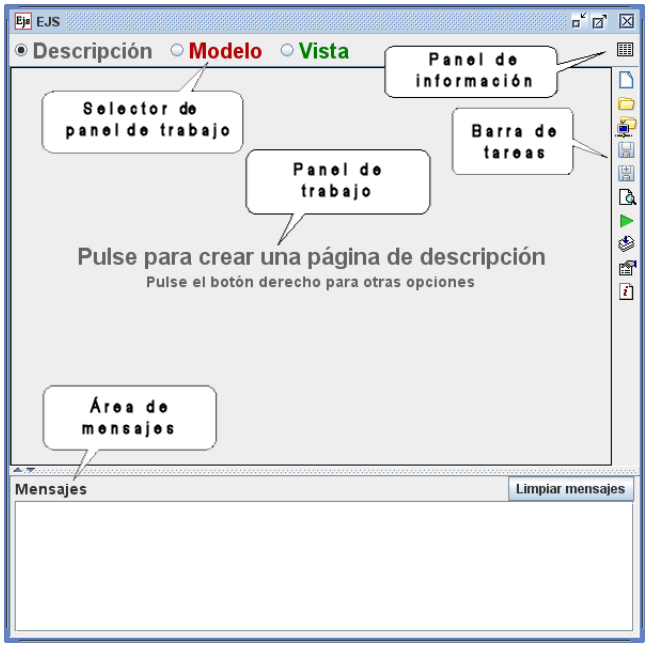

Figura 1.3: Interfaz de *Easy Java Simulations* con anotaciones.

La barra de tareas de la derecha muestra una serie de iconos para crear, abrir, buscar en y guardar un archivo, configurar *EJS*, mostrar información sobre el programa y ayuda. Además, posee iconos para ejecutar una simulación y para almacenar una o más simulaciones en un solo archivo comprimido de tipo jar. Haciendo clic con el botón derecho de su ratón sobre los iconos de la barra de tareas podrá elegir otras acciones alternativas

relacionadas que iremos describiendo conforme vayan siendo necesarias. La parte inferior de la interfaz contiene un área de mensajes donde *EJS* nos muestra información. La parte central de la interfaz contiene los paneles de trabajo donde se construye la simulación.

Easy Java Simulations contiene tres paneles de trabajo. El primer panel, llamado *Descripción*, nos permite crear un documento multimedia en HTML que describa nuestro modelo. Cada página aparece en una pestaña dentro del panel y, haciendo clic con el botón derecho del ratón sobre la pestaña, el usuario podrá editar la página o importar otras existentes. El segundo panel, *Modelo*, está dedicado al proceso de modelado. Usaremos este panel para crear variables que describan el modelo, para inicializar dichas variables y para escribir algoritmos que describan cómo nuestro modelo cambia con el tiempo. El tercer panel, *Vista*, está dedicado a la tarea de construir la interfaz gráfica para el usuario final que permitirá controlar la simulación y mostrar sus salidas. Construiremos la interfaz eligiendo elementos de las distintas paletas y añadiéndolos a la vista del *árbol de elementos*. Por ejemplo, la paleta de *Interfaz* contiene botones, barras de desplazamiento y campos de texto, y la paleta de *Elementos de dibujo 2D* contiene elementos para trazar datos en dos dimensiones.

1.3. INSPECCIONANDO LA SIMULACIÓN

Para entender cómo los paneles de trabajo, *Descripción*, *Modelo* y *Vista*, funcionan coordinadamente, vamos a inspeccionar y ejecutar una simulación ya existente. Las salidas por pantalla no pueden sustituir a una demostración en vivo por lo que le animamos a que realice las acciones en su ordenador conforme avance en la lectura.

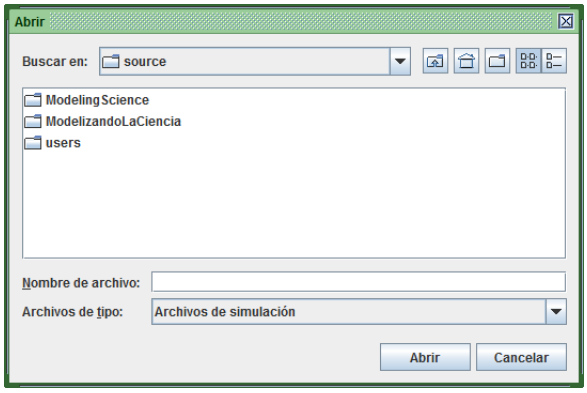

Figura 1.4: El cuadro de diálogo *Abrir* le permite buscar en su disco duro y cargar simulaciones existentes.

Haga clic en el icono *Abrir* en la barra de tareas de *EJS* y apa-

recerá un cuadro de diálogo parecido al que se muestra en la Figura 1.4 mostrándonos el contenido del directorio **source** de su espacio de trabajo. Vaya al directorio **ModelizandoLaCiencia** y abra el subdirectorio **Cap02_Intro** allí encontrará un archivo llamado **MasaYMuelle.xml**. Seleccione este archivo y haga clic en el botón *Abrir* de la ventana.

¡Ahora las cosas cobran vida! *EJS* lee el documento **MasaYMuelle.xml** a partir de cuyo contenido rellena los paneles de trabajo y aparecen dos nuevas ventanas de *EJS* en su pantalla, como muestra la Figura 1.5. Una advertencia: Los objetos pueden ser arrastrados sobre la ventana de la maqueta pero esto determinará las condiciones iniciales. Normalmente es mejor determinar las condiciones elementales utilizando la tabla de variables que se describe en la Sección 1.3.2.

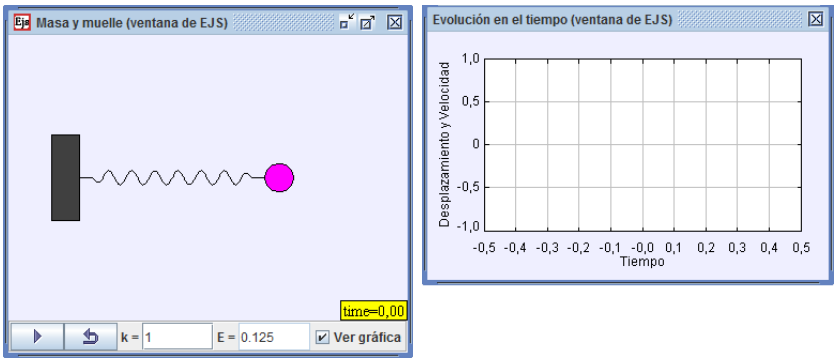

Figura 1.5: Ventana de la maqueta de *EJS* para la simulación **Masa y Muelle**. El título de la ventana muestra que son ventanas de *EJS* y que el programa no está ejecutándose.

> Los lectores más impacientes o precoces puede que hayan intentado hacer clic sobre el botón verde de ejecución ▶ de la barra de tareas para ejecutar nuestro ejemplo antes de continuar con este tutorial. Aquellos lectores que hayan hecho esto no estarán interactuando con *EJS* sino con un programa Java compilado y ejecutado. Salga de la ejecución del programa cerrando la ventana *Masa y muelle* o haciendo clic con el botón derecho en el botón de ejecución ▶(ahora en color rojo) antes de continuar.

1.3.1. El panel de la *Descripción*

Seleccione el panel de *Descripción* haciendo clic sobre el selector en la parte superior de la ventana de *EJS* y podrá ver dos páginas de narración sobre esta simulación. La primera página, que se muestra en la Figura 1.6, contiene una breve explicación sobre el modelo de masa y muelle. Haga clic en la pestaña *Actividades* para ver la segunda página.

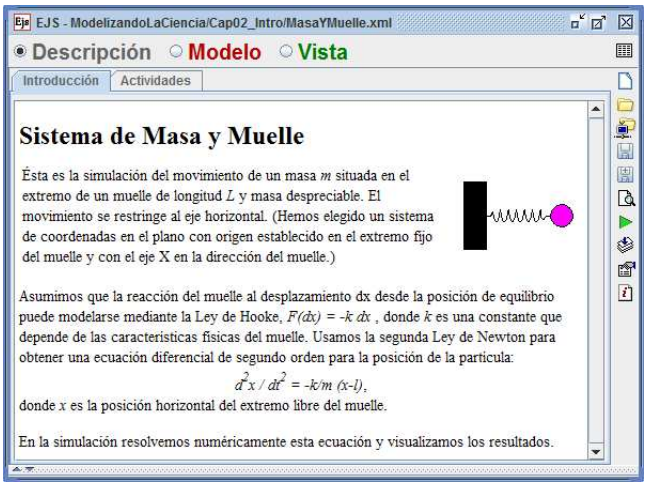

Figura 1.6: Páginas de descripción para la simulación de la masa y el muelle. Haga clic en una pestaña para mostrar la página correspondiente. Haga clic con el botón derecho para editarla.

Una *Descripción* es un texto multimedia en formato HTML que proporciona información e instrucciones sobre la simulación. HTML significa Lenguaje de Marcas de Hipertexto (HyperText Markup Language) y es el protocolo más usado para dar formato a los documentos de las páginas Web. *EJS* proporciona un sencillo editor de HTML que permite crear y modificar páginas desde *EJS*. También se pueden importar páginas haciendo clic con el botón derecho sobre una de las pestañas del panel de trabajo (Ver Sección 1.6.3.) Las páginas de descripción son una parte esencial del proceso de modelado y estas páginas se distribuyen junto con el modelo compilado cuando el modelo es distribuido como una aplicación Java o publicado en un servidor Web como un subprograma (applet). Estas opciones de distribución se describen en el Apéndice **??**.

1.3.2. El panel del *Modelo*

El panel de trabajo *Modelo* es donde se define el modelo de modo que pueda ser convertido en un programa por *EJS*. En esta simulación, estudiaremos el movimiento de una partícula de masa m unida a uno de los extremos de un muelle de masa despreciable y de longitud en equilibrio L. El muelle está unido a una pared por su otro extremo y se mueve estrictamente en horizontal. Aunque la masa oscilante tiene una solución analítica bien conocida, es útil empezar con un modelo del oscilador armónico simple con el fin de poder comparar nuestra solución numérica con el resultado analítico exacto.

Nuestro modelo asume que se producen únicamente oscilaciones pequeñas, de modo que el muelle responde a un desplazamiento (horizontal), δx, desde su posición de equilibrio, L, con una fuerza dada por la ley de Hooke, $F_x = -k\,\delta x$, donde k es la constante de elasticidad del muelle que depende de las características físicas del mismo. Usamos la segunda ley de Newton para obtener una ecuación diferencial de segundo orden para la posición de la partícula:

$$\frac{d^2\,x}{dt^2} = -\frac{k}{m}\,(x - L). \tag{1.3.1}$$

Observe que usamos un sistema de coordenadas con el eje X situado a lo largo del muelle y con origen en el extremo fijo del muelle. La partícula se encuentra en la posición x y su desplazamiento desde la posición de equilibrio, $\delta x = x - L$, es cero cuando $x = L$. Resolvemos este sistema numéricamente para estudiar cómo evoluciona el modelo con el paso del tiempo.

Vamos a examinar cómo hemos implementado el modelo de masa y muelle eligiendo el selector *Modelo* e inspeccionando cada uno de sus cinco paneles.

1.3.2.1. Declaración de variables

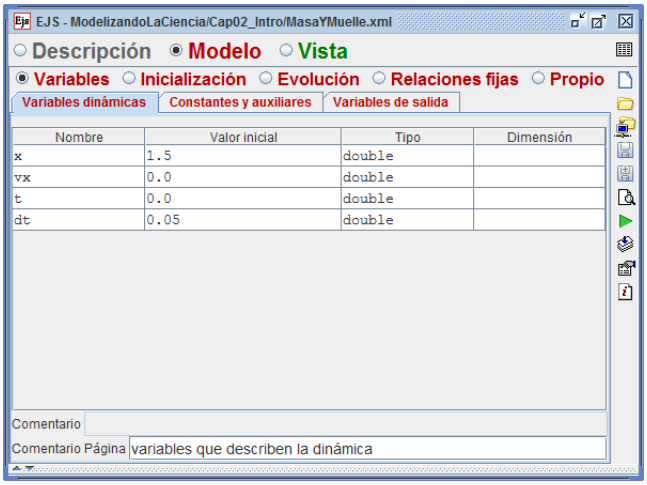

Figura 1.7: El panel de trabajo *Modelo* contiene cinco subpaneles. Se muestra el subpanel para la definición de las variables dinámicas del ejemplo. Otras pestañas en este subpanel definen variables adicionales tales como la longitud del muelle L y la energía E.

Cuando implementamos un modelo, un buen primer paso es identificar,

definir e inicializar las variables que describen nuestro sistema. El término *variable* es muy general y se refiere a cualquier cosa a la que podemos dar un nombre, incluyendo una constante física o un gráfico. La Figura 1.7 nos muestra una tabla de variables en *EJS*. Cada fila define una variable del modelo especificando para ella un nombre, un tipo, una dimensión y un valor inicial.

Las variables en los programas de ordenador pueden ser de varios tipos dependiendo de la información que guardan. Los tipos más frecuente son `boolean` que se utiliza para los valores verdadero/falso, `int` para enteros, `double` para números de alta precisión ($\approx$ 16 dígitos significativos) y `String` para textos. Usaremos los distintos tipos a lo largo del libro pero el modelo de masa y muelle sólo utiliza variables de tipo double y boolean.

Las variables pueden ser utilizadas como parámetros, variables de estado, o para entradas y salidas del modelo. Nosotros hemos declarado una variable para el tiempo, `t`, otra para la posición de la partícula, `x`, y otra para la velocidad en la dirección del eje X, `vx`. Además hemos definido variables que no aparecen en (1.3.1). La razón de estas variables auxiliares, como la energía cinética, la energía potencial y la energía total, se aclarará más adelante. En la parte inferior del panel de variables encontramos un campo para comentarios que proporciona una descripción del papel de cada variable en el modelo. Haciendo clic en cada una de las variables podremos ver el comentario correspondiente.

1.3.2.2. Inicialización del modelo

Establecer correctamente las condiciones iniciales del modelo es importante en su implementación ya que el modelo ha de comenzar en una situación físicamente posible. Nuestro modelo es relativamente simple y lo inicializamos con valores enteros (o expresiones simples en Java como `0.5*m*vx*vx`) en la columna *Valor inicial* de la tabla de variables y *EJS* utiliza dichos valores cuando se inicia la simulación.

> Modelos más avanzados pueden requerir un algoritmo de inicialización. Por ejemplo, un modelo molecular dinámico debe establecer las velocidades de un conjunto de partículas. El panel de *Inicialización* nos permite definir una o más páginas de código Java que realicen el cálculo necesario. *EJS* convierte este código en un método Java[2] y llama a este método al empezar y siempre que la simulación sea reiniciada. El panel de la *Inicialización* del modelo de masa y muelle no se muestra aquí porque está vacío. Vaya a la Sección 1.3.2.4

[2]Un método Java es algo parecido a una función o una subrutina en lenguajes de programación procedimental.

para ver un ejemplo de cómo aparece en *EJS* el código Java.

1.3.2.3. La evolución del modelo

El panel de *Evolución* nos permite escribir código Java que determine cómo el sistema masa-muelle se desarrolla en el tiempo y usaremos esta opción frecuentemente para modelos no basados en ecuaciones diferenciales ordinarias (EDOs). Hay sin embargo una segunda opción que nos permite introducir ecuaciones diferenciales ordinarias, como (1.3.1), sin utilizar programación. *EJS* proporciona un editor dedicado a esta tarea que nos permite especificar ecuaciones diferenciales en un formato que utiliza la notación matemática y que genera automáticamente el código Java necesario.

Veamos cómo el editor de ecuaciones diferenciales funciona para nuestro modelo de masa y muelle. Como los algoritmos de resolución de EDO resuelven sistemas de primer orden, una ecuación de orden superior, como (1.3.1), debe ser reducida a un sistema de primer orden. Conseguimos ésto estableciendo la velocidad como una variable independiente que obedece a su propia ecuación:

$$\frac{d\ x}{dt} = v_x \tag{1.3.2}$$

$$\frac{d\ v_x}{dt} = -\frac{k}{m}\,(x - L). \tag{1.3.3}$$

La necesidad de una ecuación diferencial adicional explica por qué habíamos declarado la variable `vx` en nuestra tabla de variables

Haciendo clic en el panel de *Evolución* se muestra el editor de EDO que aparece en la Figura 1.8.

Nótese que el editor de EDO muestra las ecuaciones (1.3.2) y (1.3.3) (usando el símbolo * para denotar la multiplicación). Los campos en la parte superior del editor especifican la variable independiente `t` y la variable incremento `dt`. Los algoritmos numéricos aproximan la solución exacta a la ecuación avanzando el estado en pasos discretos donde el incremento determina el tamaño del paso. El botón *Prelim* en la parte superior derecha del editor nos permite introducir un código preliminar que se encargue de los cálculos previos a la evaluación de las ecuaciones (circunstancia que se precisa en situaciones más complejas que la que tratamos en este ejemplo). Un menú desplegable en la parte inferior del editor nos permite seleccionar el algoritmo numérico de resolución de EDOs a emplear, el cual avanza la solución desde el valor actual para el tiempo, `t`, al siguiente valor, `t + dt`. El campo de eventos de la parte inferior del panel nos dice que no hemos

Figura 1.8: El panel de evolución de EDO mostrando la ecuación diferencial del modelo de masa y muelle y el algoritmo numérico.

definido ningún evento para esta ecuación diferencial. Podemos encontrar ejemplos con código preliminar y eventos en el Capítulo **??**.

La parte izquierda del panel de trabajo *Evolución* incluye campos para determinar cómo de suave y rápidamente queremos que funcione la simulación. El número de *imágenes por segundo* (*IPS*) especifica cuántas veces por segundo queremos que nuestra simulación sea repintada en la pantalla. El número de *pasos por visualización* (*PPV*) especifica cuántas veces (pasos) queremos que avance el modelo antes de repintar. El valor actual de `20` imágenes por segundo produce una animación fluida que, junto con el valor por defecto de un paso por visualización y el valor de `0.05` para `dt`, resulta en una simulación que funciona (aproximadamente) en tiempo real. Nosotros utilizaremos casi siempre la elección por defecto de un paso por visualización. Sin embargo, hay situaciones donde la salida gráfica del modelo consume una cantidad significativa de capacidad de proceso y necesitaremos acelerar los cálculos numéricos. En este caso incrementaremos el número de pasos por visualización de forma que el modelo avance varias veces antes de que se vuelva a dibujar la visualización. La casilla de *Arranque* indica si la simulación debe ponerse en marcha cuando el programa comience. En este caso, la casilla no está marcada así que podremos cambiar las condiciones iniciales antes de que empiece la evolución.

El panel de *Evolución* maneja los aspectos técnicos del modelo de masa y muelle sin necesidad de programar. La simulación avanza el estado del sistema en el tiempo mediante métodos numéricos de resolución de ecuaciones diferenciales utilizando el algoritmo del punto medio. El algoritmo pasa

del estado actual en el momento `t` a un nuevo estado en un nuevo momento `t + dt` antes de que la visualización se redibuje. La simulación repite esta evolución `20` veces cada segundo en ordenadores de una capacidad de procesamiento modesta. La simulación podría fluir más lenta y no tan suave en ordenadores con una capacidad insuficiente o si el ordenador está ocupado en otra tarea, pero no suele fallar.

> El modelo de masa y muelle puede resolverse con un simple algoritmo para EDOs. Nuestra librería de métodos numéricos contiene algoritmos mucho más sofisticados y *EJS* puede aplicar dichos algoritmos a sistemas grandes con ecuaciones diferenciales vectoriales con o sin puntos de discontinuidad.

1.3.2.4. Relaciones entre variables

No todas las variables del modelo se calculan utilizando algoritmos del panel de Evolución. Algunas variables pueden ser también calculadas después de que la evolución se haya aplicado. Nos referiremos a las variables que se calculan utilizando el algoritmo de evolución como variables de estado o variables dinámicas, y nos referiremos a las variables que dependen de esas variables como variables auxiliares o de salida. En nuestro modelo la energía cinética, potencial y la energía total del sistema son variables de salida porque se calculan a partir de las variables de estado.

$$T = \frac{1}{2} m {v_x}^2, \tag{1.3.4}$$

$$V = \frac{1}{2} k (x - L)^2, \tag{1.3.5}$$

$$E = T + V. \tag{1.3.6}$$

Decimos entonces que existen *relaciones fijas* entre las variables del modelo. El panel de *Relaciones fijas*, mostrado en la Figura 1.9, se usa para escribir las relaciones entre variables. Nótese qué fácil es convertir (1.3.4) a (1.3.6) a la sintaxis de Java. Asegúrese siempre de utilizar el símbolo * para la multiplicación y de escribir un punto y coma al final de cada sentencia.

Puede que se pregunte por qué no escribimos las expresiones correspondientes a relaciones fijas añadiendo una segunda página de código después de la página EDO en el panel de *Evolución*. Después de todo las páginas de evolución se ejecutan secuencialmente y una segunda página podría actualizar correctamente la salida después de cada paso. La razón de que no se use el panel de *Evolución* es que las relaciones deben mantenerse *siempre* y hay acciones, como las hechas con el ratón, que pueden afectar a las variables indicadas. Por ejemplo, arrastrando la masa cambiamos la variable x y este

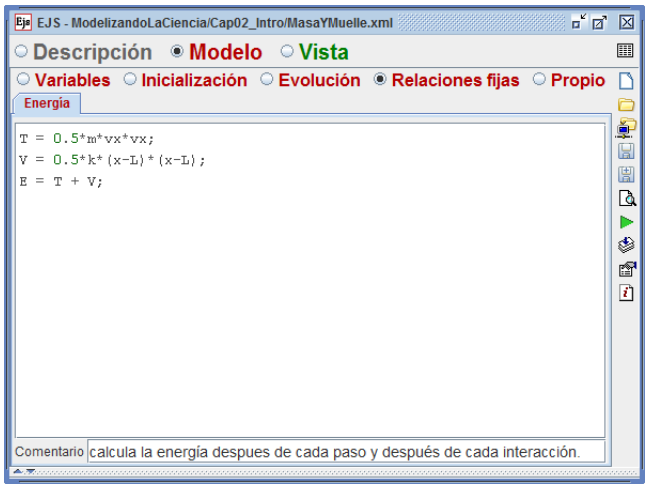

Figura 1.9: Relaciones fijas en el modelo de masa y muelle.

cambio afecta a la energía. *EJS* automáticamente evalúa las relaciones después de la inicialización, después de cada paso y si hay alguna interacción del usuario con la interfaz de la simulación. Por esta razón, es importante que las relaciones fijas se escriban en el panel destinado a ello.

1.3.2.5. Páginas Propio

En el panel de trabajo *Modelo* hay un quinto panel llamado *Propio*. Este panel puede usarse para definir métodos Java (funciones) que pueden ser usadas a lo largo del modelo. Este panel está vacío porque el modelo actual no requiere métodos adicionales, aunque haremos uso de este panel cuando modifiquemos nuestro ejemplo de masa y muelle en la Sección 1.6. Un método propio no se usa a menos que sea llamado explícitamente desde otro panel de trabajo.

1.3.3. El panel de la *Vista*

El tercer panel de trabajo de *Easy Java Simulations* es el panel de la *Vista*. Este panel de trabajo permite crear una interfaz gráfica que incluye la visualización, la interacción del usuario y el control del programa con una cantidad mínima de programación. La Figura 1.5 muestra la vista del modelo masa y muelle. Marque el selector *Vista* para examinar cómo ha sido creada la vista.

El marco de la derecha del panel de la vista, mostrado en la Figura 1.10, contiene una colección de *elementos para la vista* agrupados según su función. Los elementos para la vista funcionan como bloques de construcción que pueden ser combinados para formar una interfaz de usuario completa. Cada elemento para la vista es un objeto especializado con una representación específica en pantalla. Para ver información sobre un elemento, haga clic en su icono y presione la tecla *F1* o haga clic con el botón derecho del ratón y seleccione *Ayuda* en el menú emergente que aparecerá. Para crear una interfaz de usuario creamos una ventana y añadimos elementos, como botones y gráficos, con sólo arrastrar y soltar, como se describe en la Sección 1.6.

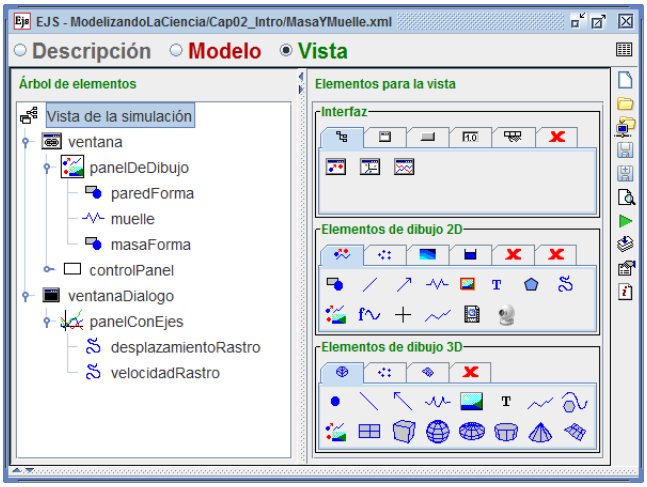

Figura 1.10: El panel de trabajo *Vista* mostrando el *Árbol de elementos* para la interfaz del modelo de masa y muelle.

El *Árbol de elementos*, mostrado a la izquierda en la Figura 1.10, muestra la estructura de la interfaz del modelo de masa y muelle. Nótese que la simulación tiene dos ventanas, una del tipo (`Ventana`) y otra del tipo (`VentanaDiálogo`), que aparecen en la pantalla de su ordenador. Estos elementos pertenecen al grupo de elementos *contenedores*, cuyo principal cometido es organizar y agrupar visualmente otros elementos de la interfaz. El árbol muestra nombres descriptivos e iconos para estos elementos. Haciendo clic con el botón derecho del ratón en los elementos del árbol aparece un menú que ayuda al usuario a modificar dicha estructura.

Cada elemento para la vista tiene un conjunto de parámetros internos llamados *propiedades* que configuran su apariencia y comportamiento. Podemos editar dichas propiedades haciendo doble clic en el elemento del árbol para mostrar una tabla a la que llamamos *inspector de propiedades*. A las propiedades de apariencia, como por ejemplo el color, se les asigna a menu-

do un valor constante, por ejemplo, `RED` (rojo, en inglés). Podemos también utilizar una variable del modelo para establecer el valor de una propiedad del elemento. Esta posibilidad de conectar una propiedad a una variable sin programar es la clave para obtener fácilmente una visualización dinámica e interactiva.

Veamos cómo se lleva a cabo este proceso en la práctica. Haga doble clic en el elemento `masaForma` (el sufijo 'Forma' que añadimos al nombre de los elementos nos ayuda a reconocer el tipo del elemento) en el árbol para mostrar su inspector de propiedades. Este elemento es la masa unida al lado que queda libre del muelle. La tabla de propiedades de este objeto aparece como se muestra en la Figura 1.11.

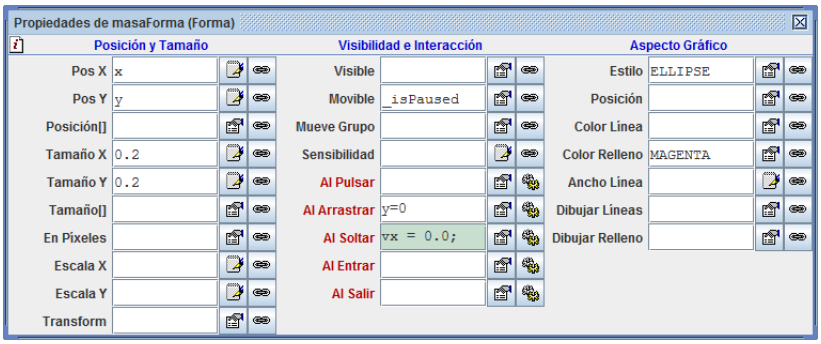

Figura 1.11: Tabla de propiedades del elemento `masaForma`.

Observe las propiedades que tienen valores constantes. El `Estilo`, `Tamaño X`, `Tamaño Y` y el `Color (de) Relleno` producen una elipse de tamaño `(0.2,0.2)` unidades (es decir, un círculo) relleno de color magenta. Mayor relevancia tiene el hecho de que las propiedades `Pos X` y `Pos Y` de la forma estén ligadas a las variables `x` e `y` del modelo. Esta simple asignación establece una conexión bidireccional entre el modelo y la vista. Estas variables cambian conforme el modelo se desarrolla y la forma sigue los valores de `x` y de `y`. Si el usuario arrastra la forma a una nueva posición, las variables `x` e `y` del modelo también cambian. Nótese que la propiedad `Movible` está sólo permitida cuando la animación está en pausa.

Los elementos pueden tener también *propiedades de acción*, que pueden estar asociadas a código Java. (Las propiedades de acción aparecen en letras rojas.) Las acciones del usuario, como arrastrar o hacer clic, llaman a sus correspondientes propiedades de acción, de modo que permiten una forma sencilla de controlar la simulación. Si el usuario arrastra la masa, el código que aparece en la propiedad `Al Arrastrar` restringe el movimiento de la forma en la dirección horizontal obligando a que la variable `y` valga

0. Finalmente, cuando se suelta el botón del ratón se ejecuta el siguiente código:

```
vx = 0.0;            // pone la velocidad a cero
_view.resetTraces(); // limpia todas las gráficas
```

Haciendo clic en el icono junto al campo de texto aparece un pequeño editor que muestra este código.

> Como el código para la acción de `Al Soltar` supera el espacio para una línea, el fondo del campo aparece más oscuro (en verde). Otros tipos de datos como las propiedades booleanas, tienen editores diferentes. Haciendo clic en el segundo icono aparece una ventana de diálogo con una lista de variables y métodos a elegir para esta propiedad.

Ejercicio 1.1. Inspectores de elementos
El inspector de propiedades para la masa muestra distintos tipos de propiedades con sus posibles valores. Explore las propiedades de la vista. Por ejemplo, los elementos `desplazamientoRastro` y `velocidadRastro` correspondientes a las gráficas del desplazamiento y de la velocidad frente al tiempo que puede encontrar en la segunda ventana de la vista. ¿Cuál es el máximo número de puntos que se pueden añadir a cada gráfica? □

1.3.4. La simulación completa

Hemos visto que *Easy Java Simulations* es una poderosa herramienta que nos permite expresar nuestro conocimiento de un modelo con un alto nivel de abstracción. Cuando hicimos el modelo de masa y muelle primero creamos una tabla de variables que describiesen el modelo e inicializamos dichas variables utilizando una columna de la tabla. Entonces usamos un panel de evolución con un editor de alto nivel para sistemas de ecuaciones diferenciales ordinarias de primer orden con el fin de especificar cómo cambiaba su estado respecto al tiempo. Después escribimos relaciones para calcular las variables auxiliares y las de salida que podían expresarse utilizando variables de estado. Finalmente, creamos la interfaz gráfica para el usuario y visualizaciones de alto nivel tan sólo arrastrando objetos desde la *Paleta* hasta el *Árbol de elementos*. Configuramos las propiedades de los elementos mediante un editor de propiedades y algunas propiedades fueron asociadas con variables del modelo.

Es importante hacer ver que las tres líneas de código en el panel de relaciones fijas (Figura 1.9) y las dos líneas de código del método de acción

de la forma son el único código explícito en Java necesario para implementar el modelo. *Easy Java Simulations* crea un programa en Java completo, procesando la información de los paneles de trabajo cuando presionamos el botón de ejecución, como se describe en la Sección 1.4.

1.4. EJECUTANDO LA SIMULACIÓN

Es el momento de ejecutar la simulación haciendo clic en el botón de *Ejecución*, ▶. *EJS* genera el código Java y lo compila, recoge archivos auxiliares y de librería y ejecuta el programa compilado. Todo con un único clic de ratón.

Al ejecutar la simulación se inicializan sus variables y se evalúan las relaciones fijas para asegurar que el modelo está en un estado consistente. La evolución respecto del tiempo del modelo empieza cuando el botón de marcha/paro de la interfaz del usuario se presiona. (El botón marcha/paro muestra el icono ▶ cuando la simulación está parada y el icono ⏸ cuando está en marcha.) En nuestro ejemplo, el programa ejecuta un método numérico para avanzar la ecuación diferencial del oscilador armónico cada 0,05 unidades de tiempo y ejecuta entonces el código de relaciones. Los datos se pasan entonces al gráfico y el gráfico se redibuja. Este proceso se repite 20 veces por segundo.

Cuando ejecutamos una simulación, *EJS* cambia el icono de ejecución a color rojo y muestra mensajes de información diciendo que la simulación se ha generado correctamente y que está funcionando. Advierta que las dos ventanas de *EJS* desaparecen y son reemplazadas por unas ventanas nuevas, aunque similares, sin el sufijo en sus títulos. Estas vistas sí que responden a las acciones del usuario. Haga clic y arrastre la partícula hasta la posición inicial deseada y entonces pulse en el botón de marcha/paro. La masa oscila alrededor de su posición de equilibrio y la gráfica muestra los datos de velocidad y desplazamiento como se muestra en la Figura 1.12.

Detenga la simulación y haga clic con el botón derecho sobre cualquiera de las áreas de dibujo de la simulación. En el menú emergente que aparece seleccione `Opciones de los elementos->panelConEjes->Herramientas de Datos` y pulse para mostrar y analizar los datos generados por el modelo. Este mismo menú ofrece otras opciones sobre la ejecución, como la captura de pantalla. Para salir del programa, cierre la ventana principal de la simulación.

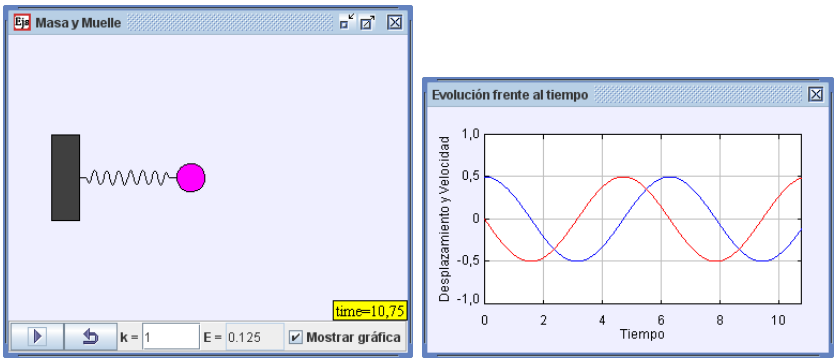

Figura 1.12: La simulación de masa y muelle muestra un dibujo interactivo del modelo y una gráfica con los datos de desplazamiento y velocidad.

1.5. DISTRIBUCIÓN DE LA SIMULACIÓN

Las simulaciones creadas con *EJS* son por sí mismas programas Java independientes que pueden distribuirse sin necesidad de que los destinatarios utilicen *EJS*. La forma más sencilla para hacer esto es empaquetar la simulación en un único archivo ejecutable de tipo jar haciendo clic en el icono *Empaquetar*, . Aparece una explorador de archivos que le permitirá elegir un nombre para el paquete jar. La dirección por defecto donde se situará este archivo es el directorio **export** de su espacio de trabajo, pero puede elegir tanto el directorio como el nombre del paquete. Este archivo independiente está preparado para ser distribuido en un CD o via Internet. Otros mecanismos de distribución están disponibles haciendo clic con el botón derecho en el icono como se describe en el Apéndice **??**.

Ejercicio 1.2. Distribución del modelo
Haga clic en el icono *Empaquetar* de la barra de tareas para crear un archivo jar independiente con la simulación de masa y muelle. Copie este archivo jar en un directorio de trabajo diferente del de su instalación de *EJS*. Cierre *EJS* y verifique que la simulación funciona como una aplicación independiente.□

Aunque el archivo jar de nuestra simulación está listo para ser usado y distribuido, un importante hecho pedagógico es que este archivo jar se ha creado de tal manera que los usuarios puedan volver a *EJS* en cualquier momento para examinar, modificar y adaptar el modelo. (Siempre y cuando se tenga *EJS* instalado, por supuesto.) El archivo jar contiene un pequeña descripción en *Lenguaje de Marcas Ampliable* (XML) de cada modelo y haciendo clic con el botón derecho en un panel de dibujo del modelo se obtiene un menú desplegable con una opción para copiar el archivo en *EJS*. Esta acción extraerá los archivos necesarios del archivo jar, buscará *EJS* en

el disco duro del usuario, copiará los archivos en la localización correcta y ejecutará *EJS* con la simulación cargada. Si ya existiera un modelo con el mismo nombre, este puede ser reemplazado. El usuario puede inspeccionar, ejecutar y modificar el modelo tal y como hemos hecho en este capítulo. Un estudiante puede, por ejemplo, conseguir un modelo o una plantilla por parte del profesor y más tarde reempaquetar el modelo una vez modificado en un nuevo archivo jar como propuesta de un ejercicio completo.

Ejercicio 1.3. Extraer un modelo
Ejecute el archivo jar independiente que contiene el modelo de masa y muelle creado en el Ejercicio 1.2. Haga clic con el botón derecho sobre la gráfica o el dibujo del modelo y seleccione el item *Abrir Modelo* EJS del menú desplegable para copiar el modelo empaquetado de vuelta a *EJS*. □

EJS ha sido diseñado para ser al mismo tiempo una herramienta de modelado y de autor, y proponemos ahora que experimente con ella para aprender cómo puede usted crear y distribuir sus propios modelos. Para empezar, recomendamos que ejecute la simulación de masa y muelle y realice las actividades de la segunda página del panel de *Descripción*. En la siguiente sección modificaremos la simulación.

1.6. MODIFICANDO LA SIMULACIÓN

Como hemos visto, una característica importante y distintiva de *Easy Java Simulations* es que nos permite crear y estudiar una simulación con un alto nivel de abstracción. En la sección anterior, hemos inspeccionado un modelo existente de una masa y un muelle, así como su interfaz de usuario. Ahora vamos a mostrar otras capacidades adicionales de *Easy Java Simulations* añadiendo fricción y una fuerza externa al modelo, así como la visualización del espacio de fases del sistema.

1.6.1. Extendiendo el modelo

Podemos añadir rozamiento en nuestro modelo introduciendo una fuerza viscosa (ley de Stoke) que es proporcional al opuesto de la velocidad $F_f = -b\, v_x$, donde b es el coeficiente de rozamiento. Tambien añadimos una fuerza externa, dependiente del tiempo, que toma la forma de una función sinusoidal $F_e(t) = A\, \sin(\omega\, t)$. La inclusión de estas fuerzas cambia la ecuación diferencial de segundo orden (1.3.1) a:

$$\frac{d^2\, x}{dt^2} = -\frac{k}{m}\,(x - L) - \frac{b}{m}\,\frac{d\, x}{dt} + \frac{1}{m}\,F_e(t), \tag{1.6.1}$$

o, como en las ecuaciones (1.3.2) y (1.3.3),

$$\frac{d\ x}{dt} = v_x, \tag{1.6.2}$$

$$\frac{d\ v_x}{dt} = -\frac{k}{m}\,(x - L) - \frac{b}{m}\,v_x + \frac{1}{m}\,F_e(t). \tag{1.6.3}$$

1.6.1.1. Añadiendo variables

Introducir nuevas fuerzas requiere que añadamos variables para el coeficiente de fricción dinámica y para la amplitud y frecuencia de la fuerza externa sinusoidal. Vuelva al panel del *Modelo* de *EJS* y seleccione su panel de *Variables*. Haga clic con el botón derecho en la pestaña de la página de variables existente para ver su menú desplegable, como en la Figura 1.13. Seleccione *Añadir una página*. Introduzca como nombre de la nueva tabla `Fricción y fuerza` y aparecerá una tabla vacía.

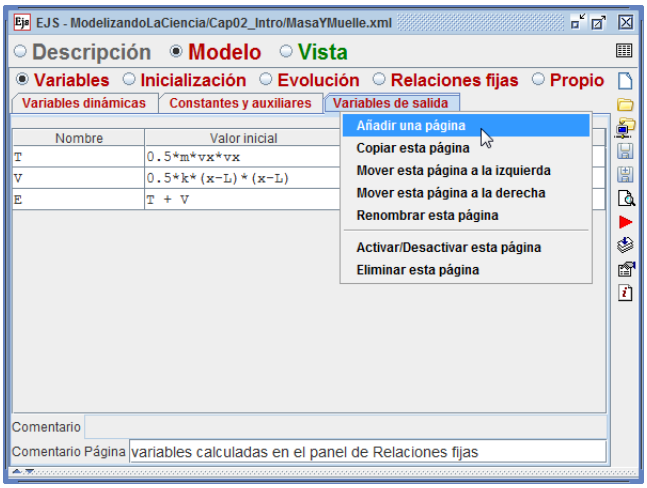

Figura 1.13: El menú desplegable de una página de variables.

Ahora usamos la nueva tabla para declarar las variables que necesitamos. Podríamos haber usado una de las tablas ya existentes, pero declararlas en varias páginas nos ayuda a organizar mejor las variables por categorías. Haga doble clic en una de las celdas de la tabla para hacerla editable y muévase a través de la tabla utilizando las flechas o el tabulador. Escriba `b` en la celda destinada al *Nombre* de la primera fila, e introduzca el valor `0.1` en la celda *Valor inicial* justo a su derecha. No necesitamos hacer nada más, ya que el *Tipo* elegido ya es correcto. *EJS* comprueba la sintaxis del valor introducido y lo evalúa. Si introducimos un valor erróneo, el fondo de la celda se mostrará en color rosa.

Nótese que cuando rellene el nombre de una nueva variable aparece una nueva fila automáticamente. Proceda de forma similar para declarar una nueva variable para la amplitud de la fuerza externa (`amp`) con valor `0.2` y para su frecuencia (`frec`) con valor `2.0`. Puede dejar una explicación sobre el significado de estas variables escribiendo un pequeño comentario para cada una de ellas en la parte inferior de la tabla. Nuestra nueva tabla de variables se muestra en la Figura 1.14. Puede ignorar la fila vacía del final de la tabla o eliminarla haciendo clic con el botón derecho sobre esta fila y seleccionando *Eliminar* del menú desplegable que aparece.

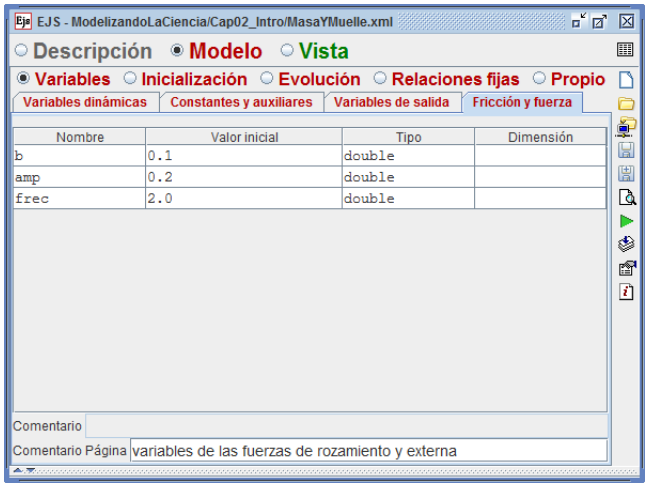

Figura 1.14: La nueva tabla de variables para los términos del rozamiento y la fuerza externa.

1.6.1.2. Modificando la evolución

Ahora, vamos a modificar la ecuación diferencial de la página de evolución añadiendo expresiones para los nuevos términos de la ecuación (1.6.3). Vaya al panel de *Evolución*, haga doble clic en la celda de *Derivada* de la segunda ecuación y escriba:

```
-k/m * (x-L) - b*vx/m + fuerza(t)/m
```

Dése cuenta de que utilizamos un método (función) llamado `fuerza` que no ha sido definido todavía. Podríamos haber escrito directamente una expresión explícita para la función sinusoidal. Sin embargo, definimos un método fuerza para que quede un código más limpio y fácil de leer y que además nos permita discutir ahora los métodos propios.

1.6.1.3. Añadiendo métodos propios

El método `fuerza` se define usando el panel *Propio* del modelo. Vaya a este panel y haga clic en el área central que estará vacía para crear una nueva página de código propio. Llame a la página *fuerza*. Notará que la página se crea con un código plantilla que define el método. Edite el código para que se lea:

```
public double fuerza (double tiempo) {
  return amp*Math.sin(frec*tiempo); // fuerza externa sinusoidal
}
```

Teclee este código exactamente como se muestra aquí, incluyendo las mayúsculas. El compilador producirá errores al generar la simulación si hay cualquier error de sintaxis.

Observe que estamos pasamos al método `fuerza`, como parámetro de entrada, el valor del tiempo que queremos utilizar para calcular la fuerza externa. Pasarle el valor del tiempo es importante. Sería incorrecto pedirle al método que usase el valor de la variable t como sigue:

```
public double fuerza () {
  return amp*Math.sin(frec*t);// implementación incorrecta del método
}
```

La razón de que debamos pasar el tiempo como parámetro es que el tiempo cambia a lo largo del paso de la evolución. Para que el algoritmo de resolución de la ecuación diferencial calcule correctamente la fuerza dependiente del tiempo en instantes intermedios de un paso de integración numérica, el tiempo debe pasarse al método que calcula la derivada.

> Las variables que cambian (evolucionan) deben pasarse a los métodos que se usan en el cálculo de la derivada porque los métodos numéricos evalúan la columna *Derivada* en el panel de trabajo de las EDO en valores intermedios entre t y $t + dt$. (véase el capitulo **??**.) En otras palabras, la variable independiente y cualquier otra variable dinámica que aparezca en la columna *Estado* del editor de EDO debe pasarse a cualquier método al que se haga referencia en la columna *Derivada*. Aquellas variables que permanecen constantes durante el paso de evolución podrían utilizarse sin ser pasadas por parámetros de entrada ya que se puede utilizar el valor que tenía la variable al inicio del paso.

1.6.2. Mejorando la Vista

Ahora vamos a añadir a la *Vista* una visualización del espacio de fases (desplazamiento frente a velocidad) de la evolución del sistema. También vamos a añadir campos de entrada para mostrar y modificar el valor de los parámetros de rozamiento, amplitud y frecuencia.

Vaya al panel de la *Vista* y fíjese que la paleta de *Interfaz* contiene varios subpaneles. Haga clic en la pestaña con el icono para mostrar la paleta *Ventanas, contenedores y paneles de dibujo*. Haga clic en el icono de Panel con ejes, , en esta misma paleta. Puede mantener el cursor sobre cualquiera de los iconos para que se muestre una nota que describe el elemento brevemente en caso de que tenga dificultades para reconocer el icono. Al seleccionar un elemento, éste muestra un marco de color alrededor del icono en la paleta y el cursor se convierte en una varita mágica, . Estos cambios indican que *EJS* está listo para crear un elemento del tipo seleccionado.

Haga clic en el elemento `ventanaDialogo` en el *Árbol de elementos* como muestra la Figura 1.15 para añadir un panel con ejes a la vista.

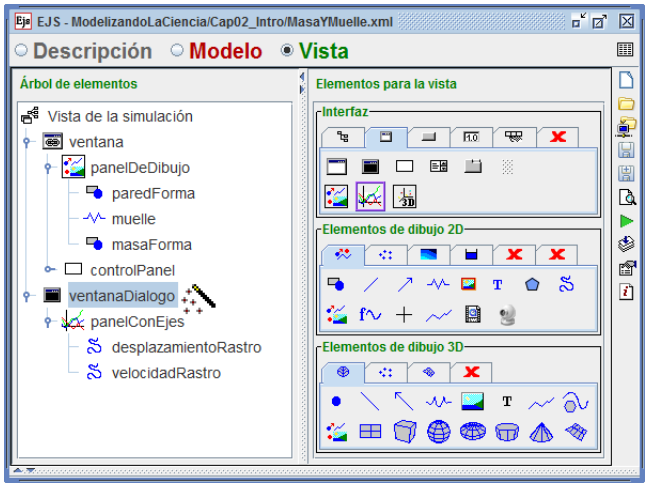

Figura 1.15: Creación de un panel con ejes como hijo del elemento `diálogo` de la vista.

EJS nos pide un nombre para el nuevo elemento y entonces lo crea como hijo del ya existente, `ventanaDialogo`. Aparece una nueva gráfica, pero la ventana de diálogo es demasiado pequeña. Vuelva al modo de diseño (líbrese de la varita mágica) haciendo clic en cualquier área en blanco del *Árbol de elementos* o presionando la tecla *Esc*. Puede reescalar la ventana arrastrando su esquina o haciendo doble clic en el elemento `ventanaDialogo`

en el árbol para mostrar su tabla de propiedades y cambiar alli su `Tamaño` a `"385,530"` de forma que doblará su altura. Finalmente, edite la tabla de propiedades del panel con ejes recién creado para cambiar el `Título` a `Espacio de fases`, el `Título X` a `Desplazamiento` y el `Título Y` a `Velocidad`. (*EJS* añadirá comillas a estas cadenas de texto para adecuarse a la sintaxis de Java.) Seleccione el mínimo y máximo para las escalas X e Y en `-1` y `1`, respectivamente, y deje las otras propiedades tal y como están.

El panel con ejes es un contenedor para la gráfica del espacio de fases. Los datos del espacio de fases se dibujan en este panel utilizando un elemento del tipo `Rastro`, . Encuentre el elemento `Traza` en la `Paleta de Elementos de Dibujo 2D` y cree un elemento de este tipo haciendo clic con la varita mágica sobre el panel de espacio de fases. Finalmente, edite las propiedades del nuevo elemento para establecer su `Entrada X` a `x - L` y su `Entrada Y` a `vx`. Estas asignaciones hacen que la simulación añada un nuevo punto `(x - L,vx)` a la traza después de cada paso de evolución, de forma que se dibuje la gráfica de espacio de fases que se muestra en la Figura 1.16.

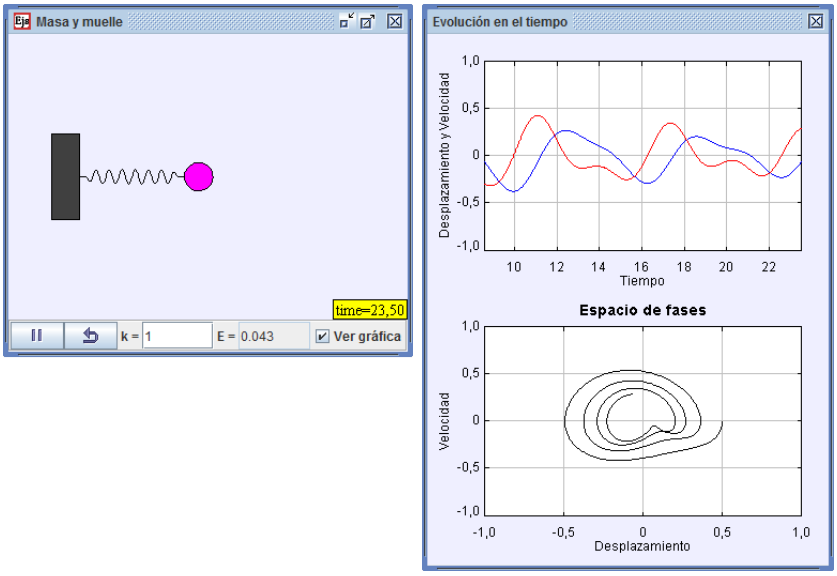

Figura 1.16: La simulación modificada. La ventana de diálogo incluye ahora una gráfica frente al tiempo y una gráfica del espacio de fases.

Para terminar con las modificaciones, vamos a añadir un nuevo panel en la parte superior de la imagen que muestre los parámetros de la fuerza externa sinusoidal.

- Seleccione el icono de `Panel`, ▭, en el grupo de *Ventanas, contenedores y paneles de dibujo* dentro de la paleta *Interfaz*. Haga clic con la varita

mágica en el elemento llamado `ventana` en el *Árbol de elementos* para crear un nuevo panel llamado `paramFuerzaPanel` en la parte superior de la ventana. Utilice el inspector de propiedades para elegir el diseño del panel como *FLOW:center,0,0* y su tipo de borde como `LOWERED_ETCHED`.

- Seleccione el icono del elemento `Etiqueta`, **A**, en el subgrupo *Botones y decoración* de la paleta de *Interfaz* y cree un nuevo elemento de este tipo en el panel de parámetros de la fuerza. Cambie el texto de la etiqueta a `"frecuencia="`.
- Seleccione el elemento de `Campo Numérico`, 20, y cree un nuevo elemento llamado `frecCampo` en el panel de parámetros de la fuerza. Edite su tabla de propiedades como muestra la Figura 1.17. La conexión con la variable `frec` se establece usando la propiedad `Variable`. Haga clic en el segundo icono de la derecha del campo de texto de la propiedad, 🔗, y elija la variable apropiada. La lista de variables muestra todas las variables del modelo que pueden ser utilizadas en este campo. La propiedad `Formato` indica el número de cifras decimales que desea que muestre el valor de la variable.
- Repita todo el proceso para añadir la variable `amp` a la interfaz de usuario.

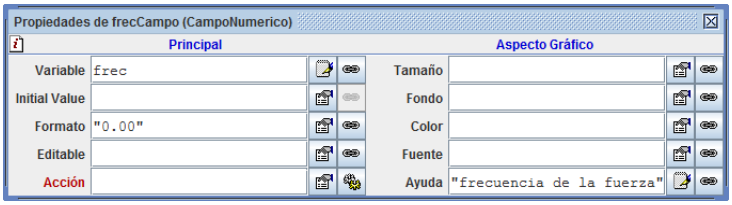

Figura 1.17: Tabla de propiedades del elemento `freqCampo`.

1.6.3. Cambiando la descripción

Ahora que hemos cambiado el modelo y su vista deberíamos modificar también las páginas de descripción de nuestra simulación. Vaya al panel de trabajo *Descripción* y haga clic con el botón derecho sobre la pestaña de la primera página, con el nombre `Introducción`, para mostrar el menú desplegable para esa página. Seleccione la opción *Editar/Ver* esta página. La página de descripción cambiará al modo de edición, tal y como muestra la Figura 1.18, y aparecerá un sencillo editor que proporciona acceso directo a las funciones más comunes de HTML.

Si prefiere usted usar su propio editor, puede copiar y pegar fragmentos de HTML desde su editor al editor de *EJS*. Si conoce la sintaxis HTML puede además editar el código fuente directamente haciendo clic sobre el icono <>

en la barra de utilidades. Puede además importar páginas HTML completas a *EJS* haciendo clic con el botón derecho sobre una de las pestañas del panel.

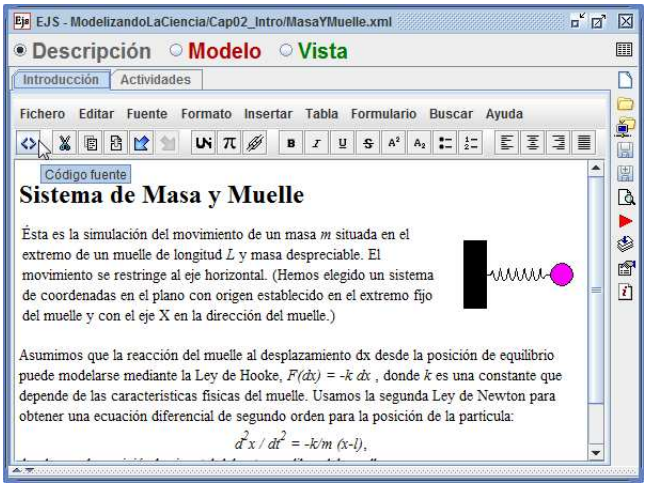

Figura 1.18: El editor de HTML de *EJS*. El cursor señala el icono para cambiar al modo de edición de código fuente.

Edite las páginas de descripción como crea conveniente. Al menos cambie la discusión del modelo para incluir el rozamiento y la fuerza externa. Cuando haya terminado, guarde la nueva simulación con un nombre diferente haciendo clic sobre el icono *Guardar Como* de la barra de tareas de *EJS*, . Cuando se le pida, introduzca un nuevo nombre para el archivo XML de su simulación. Nosotros guardamos la simulación modificada en el archivo **MasaYMuelleCompleto.xml** en el directorio de este capítulo.

1.7. BUSCANDO MODELOS

Ahora que hemos cubierto los aspectos básicos de *EJS* y ya sabe cómo cargar, inspeccionar, ejecutar y hasta modificar un ejemplo, puede que esté interesado en encontrar más ejemplos para ver qué han hecho otros usuarios con *EJS*. Es posible que pueda encontrar un modelo existente que se ajuste a sus necesidades o que pueda usted modificar fácilmente para usarlo en sus clases.

Hay dos sitios donde puede usted mirar en busca de más modelos. El primer sitio a mirar es el directorio **source** de muestra que viene con su distribución de *EJS*. En el directorio **source** del espacio de trabajo de la distribución encontrará algunos subdirectorios con simulaciones de muestra. Estos directorios de muestra se copiaron también en su propio espacio de trabajo (salvo que indicara usted lo contrario) cuando ejecutó *EJS* por primera vez.

El segundo, y quizá más interesante, lugar (de hecho, lugares) para buscar nuevos modelos está accesible a través de Internet. El icono de la barra de herramientas de *EJS* de librerías digitales, , abre una ventana que le permite conectar con repositorios de modelos de *EJS* accesibles a través de Internet. Esta ventana, que se muestra en la Figura 1.19, contiene un selector en su parte superior con la lista de librerías digitales disponibles. Seleccione una de estas librerías o haga clic en el botón *Leer el catálogo* para obtener el listado de modelos de *EJS* en ella. Todas las librerías funcionan de manera similar. Nosotros usaremos el repositorio de la librería digital **comPADRE** para ilustrar cómo se accede a ellas desde *EJS*.

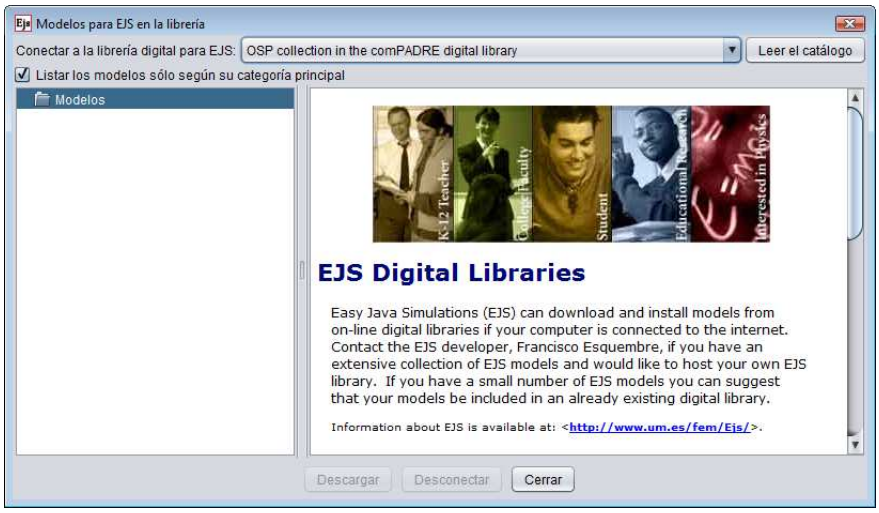

Figura 1.19: La ventana de *EJS* de librerías digitales. Seleccione uno de los repositorios disponibles usando el selector en la parte superior de la ventana, o haga clic en el botón *Leer el catálogo* para obtener el listado de modelos disponibles.

La librería **comPADRE Pathway**, que es parte de la Librería Digital de Ciencia de los Estados Unidos de América, es una red en expansión de colecciones de recursos educativos que sirven de apoyo a profesores y estudiantes de Física y Astronomía. De especial relevancia para nosotros es la colección Open Source Physics en comPADRE, disponible en *<http://www.compadre.org/OSP>*. Esta colección (en inglés) contiene recursos computacionales para la enseñanza en forma de simulaciones ejecutables y recursos curriculares que involucran a los estudiantes en actividades de física, computación y modelado por computadora. En particular, contiene modelos de *EJS* a cuyos archivos de código fuente (XML) puede accederse directamente desde *EJS* usando el icono de librerías digitales.

Si está conectado a Internet, seleccione la entrada *OSP collection on*

the comPADRE digital library del selector y *EJS* se conectará a la librería para obtener el más reciente catálogo de modelos de *EJS* en esta librería. En el momento de escribir estas líneas, hay cerca de 90 modelos organizados en diferentes categorías y subcategorías, y se espera que la colección siga creciendo. Como muestra el marco izquierdo de la Figura 1.20, la colección está organizada en categorías y subcategorías. Cuando el nombre de una subcategoría aparezca en rojo, haga doble-clic sobre él para expandir el nodo con la lista de modelos de la subcategoría. Como muchos modelos tienen clasificiones principal y secundarias, el botón selector en la parte superior, justo debajo del selector de librería, le permite decidir si quiere que los modelos se listen únicamente según su clasificación primaria, o que aparezcan en todas las categorías en las que estén clasificados (apareciendo, por tanto, más de una vez).

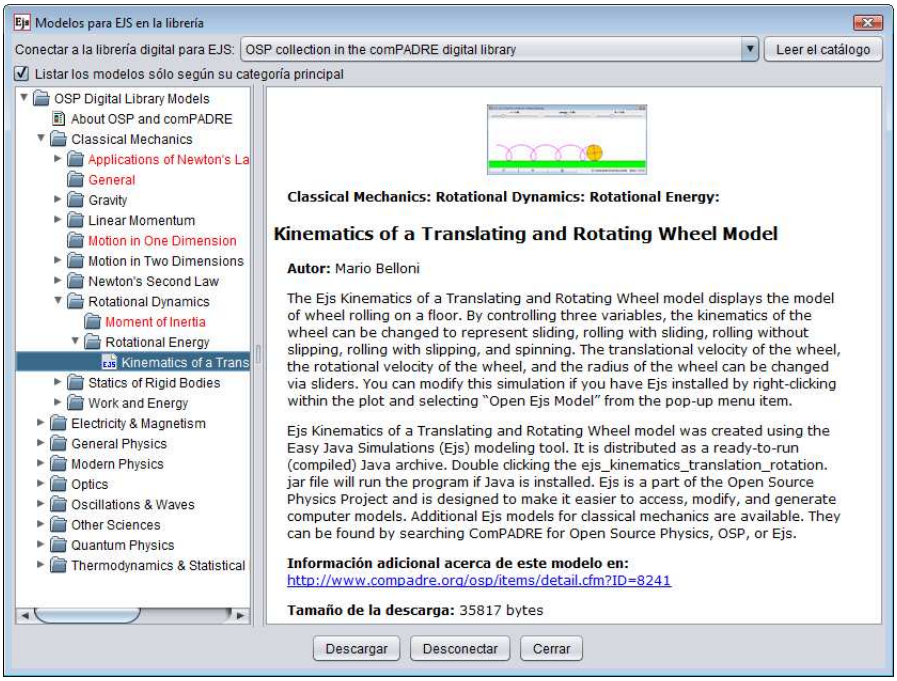

Figura 1.20: La colección OSP en la librería digital comPADRE. La colección está organizada en categorías y subcategorías. El nodo para un modelo proporciona información sobre el mismo.

Cuando hace clic en el nodo de un modelo, el marco de la derecha muestra información sobre dicho modelo obtenida de manera instantánea desde la librería. La información describe el modelo e incluye un enlace directo a la librería comPADRE para información adicional. Al hacer doble-clic en el nodo de un modelo, o al pulsar el botón *Descargar*, *EJS* obtiene los archivos del modelo y auxiliares desde la librería, le solicita un destino

en el directorio **source** de su espacio de trabajo para descargarlos, y abre el modelo en *EJS* cuando se completa la descarga. Como los ficheros fuente son típicamente pequeños, la descarga tiene lugar casi instantáneamente. Ya puede inspeccionar, ejecutar o modificar el modelo de la misma forma que hicimos anteriormente en este mismo capítulo para el modelo de la masa y el muelle.

La colección OSP de la librería digital comPADRE es un lugar altamente recomendado (si puede leer en inglés) para buscar modelos de *EJS* y material curricular de acompañamiento. Nosotros incluiremos a menudo referencias a la librería digital comPADRE en este libro, siempre que estén relacionadas con lo descrito en el texto.

1.8. RESUMEN

Este libro trata sobre el modelado y el uso de modelos para estudiar y visualizar un amplio rango de fenómenos, desde los más simples a los más complejos. Una forma adecuada de llevar a cabo este estudio es a través de simulaciones por ordenador, ésto es, usar un ordenador para obtener datos numéricos a partir de nuestros modelos conforme se desarrollan con el tiempo, y mostrar estos datos de forma que sean entendibles.

Easy Java Simulations es una herramienta de modelado y de autor expresamente dedicada a esta tarea. Ha sido diseñado para permitirnos trabajar a un alto nivel conceptual, concentrando la mayoría de nuestro tiempo y esfuerzo en los aspectos científicos de la simulación y pidiendo al ordenador que realice automáticamente todas aquellas tareas necesarias pero fácilmente automatizables. Toda herramienta, incluida *EJS*, tiene una curva de aprendizaje. La primera parte del libro contiene una serie de ejemplos detallados para que se familiarice con las posibilidades de modelado de *EJS* y con los elementos para la vista que se usan con mayor frecuencia. La segunda parte de este libro está dedicada a ejemplos más avanzados y enfatiza el contenido científico de los modelos y su comportamiento. Los apéndices cubrirán cuestiones adicionales, como una revisión de Java y una serie de pautas que le ayudarán en el inevitable momento en que cometa sus primeros errores de programación.

Modelar es tanto una ciencia como un arte. Este libro le proporciona un sólido punto de partida en la parte científica, un repaso a las técnicas requeridas por el arte y ejemplos que son útiles en la práctica.

1.9. PROBLEMAS Y PROYECTOS

Problema 1.1 (Energía)**.** Añada un tercer panel con ejes a la ventana de diálogo de la simulación **Masa y muelle completo.xml** que muestre la evolución de las energías cinética, potencial y total.

Problema 1.2 (Trazador de funciones)**.** La solución analítica para un oscilador armónico simple sin rozamiento es

$$x(t) = A\sin(w_0 t + \phi) \tag{1.9.1}$$

donde A es la amplitud (máximo desplazamiento), $w_0 = \sqrt{k/m}$ es la frecuencia natural de la oscilación y ϕ es el ángulo de fase. Consulte un libro de mecánica para encontrar la relación entre la amplitud y el ángulo de fase y el desplazamiento y velocidad iniciales. Utilice la simulación **TrazadorDeFunciones.xml** del directorio de este capítulo para comparar la solución analítica con la solución numérica generada por el modelo **MasaYMuelleCompleto.xml**.

Proyecto 1.1 (Oscilador en dos dimensiones)**.** Modifique el modelo de simulación para masa y muelle para considerar un movimiento que no esté restringido a la dirección horizontal. Asuma que un segundo muelle con constante k' produce una fuerza restauradora vertical $F_y(\delta y) = -k'\,\delta y$. Modifique la simulación para permitir al usuario especificar las constantes de la ley de Hooke, y las condiciones iniciales en ambas direcciones. Describa el movimiento que se produce sin fuerza de rozamiento pero bajo diferentes condiciones iniciales y con distintas constantes para los muelles. (Pruebe $k = 1$ y $k' = 9$.) Muestre que es posible conseguir un movimiento circular haciendo $k = k'$.

Proyecto 1.2 (Péndulo simple)**.** Cree una simulación parecida a la que se describe en este capítulo para un péndulo simple cuyo movimiento se describe mediante la ecuación diferencial de segundo orden

$$\frac{d^2\theta}{dt^2} = -\frac{g}{\ell}\sin(\theta), \tag{1.9.2}$$

donde θ es el ángulo del péndulo respecto de la vertical, g es la aceleración debida a la gravedad y ℓ es la longitud del hilo. Utilice relaciones fijas para calcular la posición x e y de la masa puntual del péndulo utilizando las ecuaciones:

$$x = \ell\,\sin(\theta)$$
$$y = -\ell\,\cos(\theta).$$

PARTE 2

From Free Fall to Chaos

Bibliografía